2

함수

함수

화살표 함수

콜백 함수

3

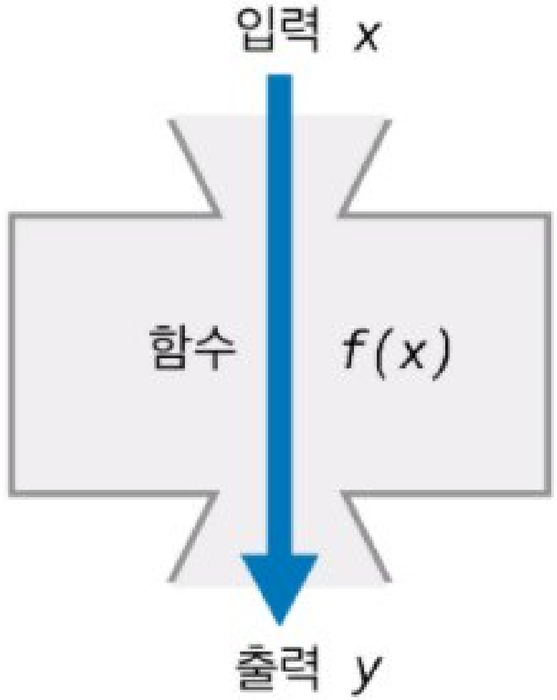
함수

특정 작업을 수행하거나 값을 계산하기 위해 설계된 재사용 가능한 코드 블록

3

함수의 흐름식

- 함수는 (컴퓨터에게 일을 시키기 위한) **지시사항의 묶음**
- 함수는 **입력** ➡️ **함수** ➡️ **출력**의 과정으로 이루어져 있음



y=f(x)

3

함수의 기본구조

```
function 함수이름(매개변수){ //function : 함수 만들기
    //매개변수 : 입력값
    실행코드
    return 결과값; //return : 결과 출력
}
```

함수가 없을때

```
var i=0;
    while (i<10){
        document.write(i);
        i+=1;
    }
    var i=0;
    while (i<10){
        document.write(i);
        i+=1;
    }
    var i=0;
    while (i<10){
        document.write(i);
        i+=1;
    }
```

```
var i=0;
while (i<10){
    document.write(i);
    i+=1;
}
var i=0;
while (i<10){
    document.write(i);
    i+=1;
}
```

3

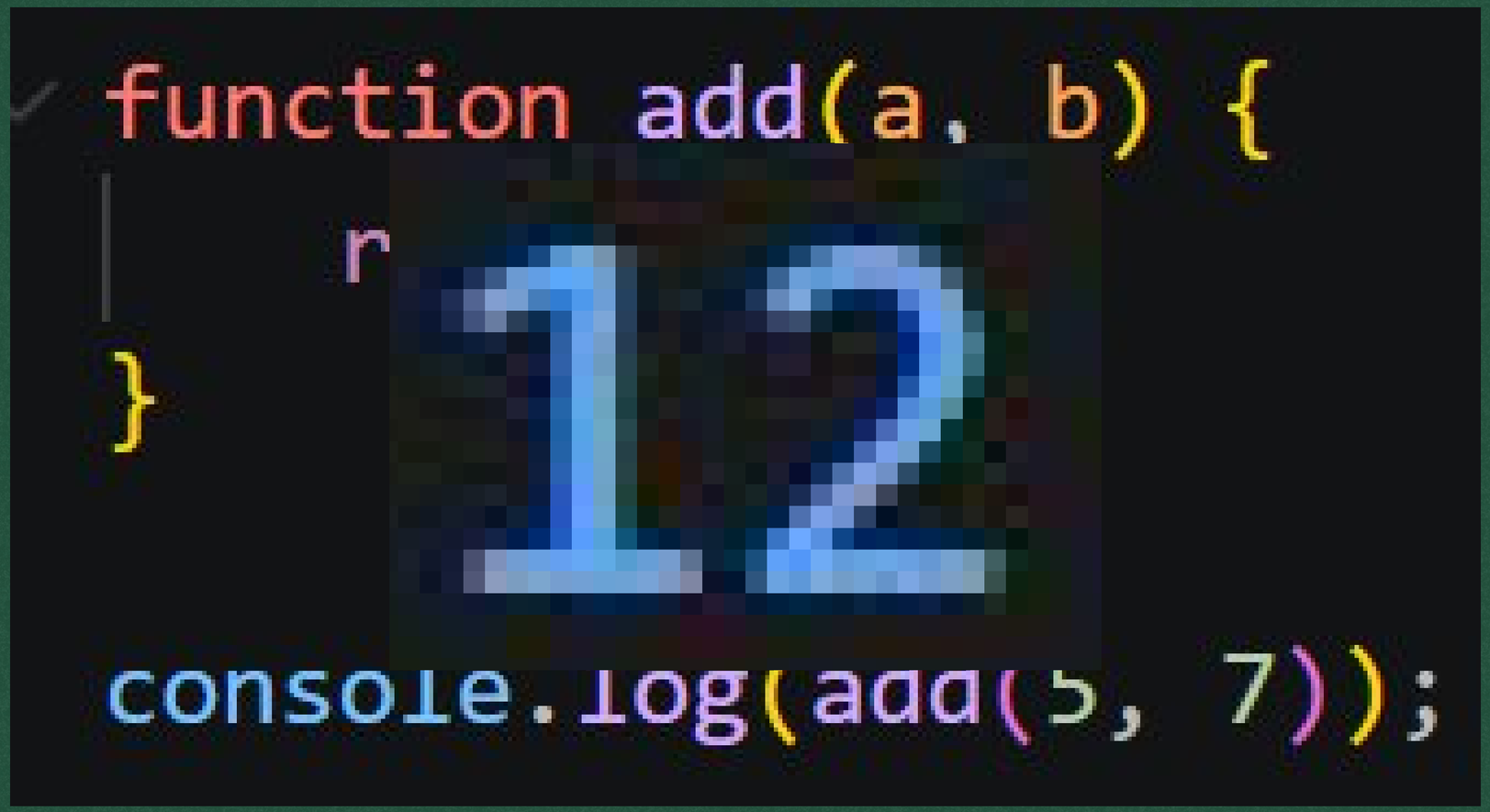
함수 예시

```
function add(a, b) {
    return a + b;
}

console.log(add(5, 7));
```

3

함수 예시



3

매개변수

매개변수는 함수 안으로 값을 전달받기 위해
사용하는 변수이다.

3

매개변수

```
function hello(name){
    console.log(name);
}
```

3

매개변수

```
function hello(name){
    console.log(name + "님 안녕하세요");
}

hello("민수");
```

3

매개변수

```
function hello(name){
    console.log(name + "님 안녕하세요");
}

hello("민수");
```

민수님 안녕하세요

3

인자

```
function hello(name){
    console.log(name + "님 안녕하세요");
}

hello("민수");
```

3

return 문

함수 종료 및 값 반환

실행 중단

3

return 문

```
function sum(a, b) {
  return a + b;
}

let result = sum(1, 2);
alert( result ); // 3
```

```
function square(x) {
    return x * x;
}

console.log(square(3));  // 9 출력
```

인자를 더하거나 곱한 후 반환합니다.

3

return 문

```
function ageCheck(age){
    if(age > 18){
        return true;
    } else{
        console.log("부모님 동의 받기");
    }
}
```

return 문을 여러 개 입력할 수 있습니다.

3

화살표 함수

화살표 함수는 함수를 더 짧고 간단하게 쓰는 방법이다.
자바스크립트에서 => 기호를 사용한다.

3

화살표 함수

```
let func = function(arg1, arg2, ...argN) {
  return expression;
};
```

```
let func = (arg1, arg2, ...argN) => expression
```

같은 동작을 합니다.

3

화살표 함수

```
let double = n => n * 2;
// let double = function(n) { return n * 2 }과 거의 동일합니다.

alert( double(3) ); // 6
```

```
let sayHi = () => alert("안녕하세요!");

sayHi();
```

3

화살표 함수

```
let age = prompt("나이를 알려주세요.", 18);

let welcome = (age < 18) ?
  () => alert('안녕') :
  () => alert("안녕하세요!");

  welcome();
```

코드를 활용할 수 있다.

3

화살표 함수

```
let sum = (a, b) => {  // 중괄호는 본문 여러 줄로 구성되어 있음을 알려줍니다.
  let result = a + b;
  return result; // 중괄호를 사용했다면, return 지시자로 결괏값을 반환해주어야 합니다.
};

alert( sum(1, 2) ); // 3
```

함수 본문이 여러 개일 때, 중괄호를 사용한다.
중괄호 사용 시, **return 지시자가 필요한다**.

3

화살표 함수

```
let sum = (a,b) => {a+b}

console.log(sum(1,2));
```

대괄호 안에 return문을 사용하지 않을 경우

```
undefined
```

3

QUIZ

함수와 매개변수 문제

매개변수로 이름과 키를 받아, 키가 **130cm** 이상이면 "[이름] 탑승 가능",
미만이면 "[이름] 탑승 불가"를 반환하는 function 키워드 함수 **checkHeight**를 작성하세요.

ex)

```
"민수 탑승 가능"
"영희 탑승 불가"
```

3

QUIZ

매개변수로 원래 수강료와 나이를 받아, 나이가 18세 이하이면 20%를 할인한 금액을 반환하고, 19세 이상이면 원래 수강료를 그대로 반환하는 화살표 함수 **getTuition**을 작성하세요.

ex
)

3

QUIZ

매개변수로 현재 마일리지와 구매 횟수를 받아,
구매 횟수가 10회 이상이면 1500점, 5회 이상이면 500점의 보너스를 더한 뒤,
최종 점수가 3000점 이상이면 "[최종점수]점 (VIP)",
아니면 "[최종점수]점 (일반)"을 반환하는 function 키워드 함수 calGoldMembership을 작성하세요.

ex) 3200점 (VIP)

3

QUIZ

매개변수로 단어(문자열)를 받아, for 반복문으로 한 글자씩 검사하여 대소문자 구분 없이 알파벳 "A" 또는 "a"의 총 개수를 숫자로 반환하는 화살표 함수 countA를 작성하세요.

ex)

```
console.log(countA("ABC"))
```

2

객체

객체

클래스

2

객체

데이터를 구조화하고 저장하는 데 사용하는 기본 단위

2

객체

```
let dog = {
    name: "초코",    // 프로퍼티

    bark: function() {    // 메서드
        console.log("멍멍!");
    }
};
```

2

객체 사용법

객체 리터럴

객체 생성자

함수 생성자

2

객체 리터럴

객체를 생성하기 위한 표기법

2

객체 리터럴

```
//객체 리터럴
const user = {
    name: "다인",
    age: 18,
    sayHello: function() {
        console.log("안녕하세요!");
    }
};

console.log(user);
```

2

객체 리터럴

```
//객체 리터럴
const user = {
    name: "다인",
    age: 18,
    sayHello: function() {
        console.log("안녕하세요!");
    }
};

console.log(user);
```

```
{name: '다인', age: 18, sayHello: ƒ}
```

2

객체 리터럴

```
//객체 리터럴
const user = {
    name: "다인",
    age: 18,
    sayHello: function() {
        console.log("안녕하세요!");
    }
};

console.log(user);
```

2

객체 리터럴

```
const user = {
  name: "다인",
  age: 18,
  sayHello: function() {
    console.log("안녕하세요!");
  }
};

user.sayHello();
```

2

객체 리터럴

```
const user = {
  name: "다인",
  age: 18,
  안녕하세요!  ;
};

user.sayHello();
```

2

객체 생성자

new Object()를 사용하여 객체를 생성하기 위한 표기법

2

객체 생성자

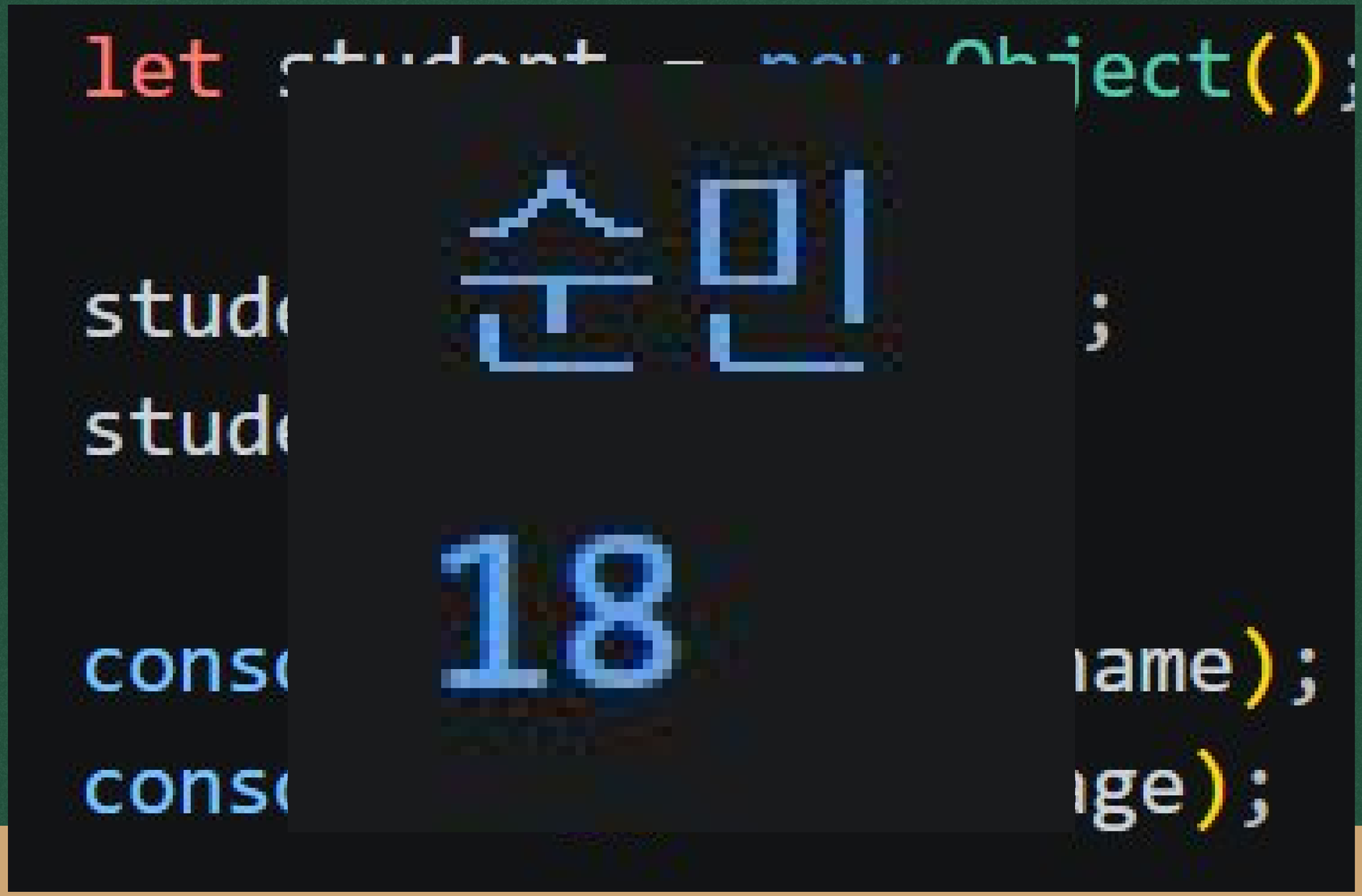
```
let student = new Object();

student.name = "순민";
student.age = 18;

console.log(student.name);
console.log(student.age);
```

2

객체 생성자



2

함수 생성자

new 함수명() 형태로 같은 구조의 객체를 여러 개 생성하기 위한 표기법

2

함수 생성자

```
function Student(name, age) {
    this.name = name;
    this.age = age;
}

let student1 = new Student("유민", 18);
let student2 = new Student("다인", 18);

console.log(student1.name);
console.log(student2.name);
```

2

함수 생성자

```
function Student(name, age) {
    this.n
    this.a
}

let studen                    유민", 18);
let studen                    나인", 18);

console.lo
console.log(student2.name);
```

객체 속성(Property)

```
const Person = {
  name: "지호",
  age: 18,
}

//조회
console.log(Person.name)
console.log(Person.age)

//추가
Person.job = "학생"
console.log(Person)
```

지호

18

{name: '지호', age: 18, job: '학생'}

```
//수정
Person.name = "순민"
Person.age = 20

console.log(Person.name)
console.log(Person.age)

//삭제
delete Person.job
console.log(Person)
```

순민

20

undefined

2

QUIZ

캐릭터의 상태를 나타내는 **hero** 객체를 만들고, 이 객체는 **name("전사")**, **hp(40)** 속성을 가지며,
호출 시 **hp**가 50 미만이면 "포션이 필요합니다!",
50 이상이면 "안전합니다!"를 출력하는 **checkStatus** 메서드를 구현하고 직접 실행하세요

2

QUIZ

카페 음료 객체를 만드는 **Drink** 생성자 함수를 만들고,
매개변수로 **menu**(메뉴명)와 **price**(가격)를 받아 저장하고,
호출 시 "아메리카노은(는) **4500**원입니다." 형태의 문자열을
콘솔창에 출력하는 **info** 메서드를 생성자 함수 내부에 구현하세요.

ex
)

```
아메리칵노은(는) 1000원입니다.
```

2

QUIZ

new Object()를 사용해 빈 객체 cart를 만든 후, item("노트북")과 count(1) 속성을 동적으로 추가하세요.
그 후 자바스크립트의 특정 키워드를 사용하여
count 속성만 완전히 삭제하고 콘솔창에 cart 객체를 출력하는 코드를 작성하세요.

2

클래스

클래스는 객체를 생성하기 위한 템플릿이다.

2

클래스

클래스

객체

2

클래스

```
class Name { // 클래스 이름의 첫 번째 글자는 대문자로 하는게 관례

    constructor(ele1, ele2) {
        this.ele1 = ele1; //this 현재 객체의 변수라는 것을 알리기 위해 사용
        this.ele2 = ele2;
    }

    methodName(element1, element2) { //메소드 사용법

    }
}
```

매서드는 있어도 없어도 상관 없음

2

this

new로 찍어낸 실제 객체(인스턴스)

2

클래스

```
// new 클래스명( );
```

```
let newOb = new Name(1, 2);
let newOb1 = new Name(3, 4);
let newOb2 = new Name(5, 6);
```

클래스 틀을 사용해 새로운 객체 생성

2

클래스

```
class Average {

    constructor(num1, num2) {
        this.ele1 = num1;
        this.ele2 = num2;
    }

    getAverage(name1, name2) {
        return `${name1}과 ${name2}의 평균은 ${(this.ele1 + this.ele2)/2}입니다`;
    }
}
```

2

클래스

다인과 유민의 평균은 85입니다

2

Static

```
class Instancecount {
let a = new Instancecount();
console.log(Instancecount.totalcount);
let c =                          ount();
let d =                          ount();
}
```

```
[Running]
4
```

2

클래스

```
class Warrior {

    constructor(name, level) {
        this.name = name;
        this.level = level;
    }

    attack() {

    }

    shield() {

    }
}
```

```
class Wizard {

    constructor(name, level) {
        this.name = name;
        this.level = level;
    }

    fireball() {

    }

    barrier() {

    }
}
```

귀찮고 같은 코드가 반복됨

2

상속

```
class Animal {

    constructor(name, age) {
        this.name = name;
        this.age = age;
    }

    sound() {
        return '';
    }
}
```

부모 클래스

2

상속

```
class Cat extends Animal {

    constructor(name, age) {
        super(name, age);
    }

    sound() {
        return `${this.name}(${this.age}살): 야옹`;
    }
}
```

extends로 자식 클래스 생성

2

값을 추가 하고 싶다면?

```
constructor(name, age, species)
    super(name, age);
    this.species = species;
}
```

2

Quiz

name과 age 값을 가지고 say라는 메소드를
가지는 People 부모 클래스를
만들고 그 클래스를 상속 받는 자식클래스를
만들어 birthday값을 추가하고
say를 오버라이딩하시오

2

Quiz

name, stock을 가진 Product라는
이름의 클래스를 만들고
totalStock을 정적 변수로하여 지금까지
생성된 모든 상품의 총재고 수량을 저장하시오